NOEH, Noah-Vincenz (nn4718)

Imperial College London

Department of Computing
Academic Year 2018-2019
Page created Thu Nov 22 19:15:32 GMT 2018

CATE

531 tk106 5
a5 nn4718 v1

0011800019886

Electronic submission

Thu - 22 Nov 2018 17:48:00

nn4718

Exercise Information

Module:	531 Prolog	**Issued:**	Wed - 14 Nov 2018
Exercise:	5 (LAB)	**Due:**	Thu - 22 Nov 2018
Title:	Syllogisms	**Assessment:**	Individual
FAO:	Kimber, Timothy (tk106)	**Submission:**	Electronic

Student Declaration - Version 1

- I declare that this final submitted version is my unaided work.

Signed: (electronic signature) Date: 2018-11-22 17:47:28

For Markers only: (circle appropriate grade)

NOEH, Noah-Vincenz (nn4718)	01562775	a5	2018-11-22 17:47:28	**A* A B C D E F**

```
Prolog Syllogisms: Summary for nn4718 of a5
-------------------------------------------

  Full Tests:
    Step 2 (syllogism/1):                 38 / 38
    Step 3 (translate/1, cl/3):           12 / 12
    Step 4 (eval/2, valid/1, invalid/1):  13 / 17
    Step 5 (test/1):                      8 / 8

```

```
%% File: syllogisms.pl
%% Name: Noah-Vincenz Noeh
%% Date: 15/11/2018
%%
%% This program is a solution to Prolog 531 Assessed Exercise 5 ’Syllogisms’
%% The exercise is to develop a parser and meta-interpreter for syllogistic
%% sentences, and use these to build a tool to determine the validity of a
%% syllogistic argument.

%% ---------------------------- Step 1 ---------------------------------------%%

%% opposite(+L, -Opp)
:- include(arguments).
:- include(utilities).
opposite([no|Tail], [some|Tail]) :- !.
opposite([some, Thing1, is, not, a, Thing2], [a, Thing1, is, a, Thing2]) :- !.
opposite([some, Thing1, is, not, Thing2], [a, Thing1, is, Thing2]) :- !.
opposite([some|Tail], [no|Tail]) :- !.
% for ’a’ and ’every’
opposite([_, Thing1, is, a, Thing2], [some, Thing1, is, not, a, Thing2]).
opposite([_, Thing1, is, Thing2], [some, Thing1, is, not, Thing2]).


%% ---------------------------- Step 2 ---------------------------------------%%

%% Stage 2.1 - This is the suggested way to develop the solution.
%% Once Stage 2.2 is complete you can delete or comment out this code.
%% syllogism/0

/*
syllogism --> article, [B], is_, optional_article, [C].
syllogism --> some, [B], is_, not, optional_article, [C].
syllogism --> no_, [B], is_, optional_article, [C].
syllogism --> some, [B], is_, optional_article, [C].
article --> [a].
article --> [every].
some --> [some].
no_ --> [no].
is_ --> [is].
not --> [not].
optional_article --> [a].
optional_article --> [].
*/

%% Stage 2.2
%% syllogism(-Clauses)

syllogism(Clauses) --> article, [B], is_, optional_article, [C], {
    Term1 =.. [C, X],
    Term2 =.. [B, X],
    Clauses = [(Term1:-Term2)]
}.

syllogism(Clauses) --> some, [B], is_, not, optional_article, [C], {
    Term1 =.. [not, C],
    Term2 =.. [some, B, Term1],
    Term3 =.. [B, Term2],
    Term4 =    (Term3:-true),
    Term5 =.. [C, Term2],
    Term6 =    (false:-Term5),
    Clauses = [Term4,Term6]
}.

syllogism(Clauses) --> no_, [B], is_, optional_article, [C], {
    Term1 =.. [B, X],
    Term2 =.. [C, X],
    Clauses = [(false:-Term1, Term2)]
}.

syllogism(Clauses) --> some, [B], is_, optional_article, [C], {
    Term1 =.. [some, B, C],
    Term2 =.. [B, Term1],
    Term3 =    (Term2:-true),
    Term4 =.. [C, Term1],
    Term5 =    (Term4:-true),
    Clauses = [Term3, Term5]
}.

article --> [a].
article --> [every].
some --> [some].
no_ --> [no].
is_ --> [is].
not --> [not].
optional_article --> [a].
optional_article --> [].


%% ---------------------------- Step 3 ---------------------------------------%%

%% translate(+N)
%:- dynamic cl/3.
translate(N) :-
        findall(Premise, p(N, Premise), List),
  write(List),
  nl,
        c(N, Conclusion),
        opposite(Conclusion, Opposite),
        get_clauses([Opposite|List], [], Phrases),
  write(Phrases),
  assertall(N, Phrases).

get_clauses([], Acc, Acc).
get_clauses([X|Tail], Acc, Phrases) :-
        phrase(syllogism(Clauses), X), % returns semantics as clauses
  append(Clauses, Acc, NewAcc),
        get_clauses(Tail, NewAcc, Phrases).

%% ---------------------------- Step 4 ---------------------------------------%%

%% eval(+N, +Calls)
eval(_, true) :- !.
eval(N, (P, Q)) :-
  !,
  eval(N, P),
  eval(N, Q).

eval(N, P) :-
  cl(N, P, Body),
  eval(N, Body).
  %call(Z) --> Call, {translate(), cl(N, Call, B), eval(N, B)}.


%% valid(?N)

valid(N) :-
  eval(N, false).


%% invalid(?N)

invalid(N) :-
  \+ eval(N, false).

```

```

%% ---------------------------- Step 5 ---------------------------------------%%

%% test(+N)


test(N) :-
  write('syllogism '), write(N), write(':'),
  nl,
  findall(Premise, p(N, Premise), PremiseList),
  printList(PremiseList),
  nl,
  write('   =>'),
  nl,
  c(N, Conclusion),
  write('   '),
  printListElems(Conclusion),
  nl,
  nl,
  write('Premises and opposite of conclusion converted to clauses:'),
  nl,
  show_clauses(N),
  nl,
  print_validity(N).

printList([X,Y]) :-
  write('   '),
  printListElems(X),!,
  nl,
  write('   '),
  printListElems(Y).

printList([X]) :-
  printListElems(X).

printListElems([X|Tail]) :-
  write(X), write(' '),
  printListElems(Tail).

printListElems([]).


print_validity(N) :-
  valid(N),!,
  write('false can be derived, syllogism '), write(N), write(' is valid.'),
  nl.

print_validity(N) :-
  write('false cannot be derived, syllogism '), write(N), write(' is invalid.'),
  nl.
```

```



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% MAC/MCSS/MRes
%% 531 Prolog
%% Assessed Exercise 5 - Syllogisms
%% arguments.pl  (Test syllogism arguments)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


p(1,[a,robin,is,a,bird]).    %    bird(X):-robin(X)
p(1,[no,bird,is,a,reptile]). %  false:-bird(X),reptile(X)




p(2,[a,human,is,a,mammal]).       % mammal(X):-human(X))
p(2,[a,mammal,is,warm_blooded]).  % warm_blooded(X):-mammal(X)




p(3,[a,human,is,a,mammal]).  % mammal(X):-human(X)
p(3,[some,human,is,deaf]).
    %  human(some(human,deaf):-true
    %  deaf(some(human,deaf)):-true


p(4,[a,robin,is,a,bird]).   % bird(X):-robin(X)
p(4,[no,bird,is,a,reptile]).  % false:-bird(X),reptile(X)



p(5,[every, philosopher, is, a, logician]).  % logician(X):-philosopher(X)
p(5,[every, philosopher, is, a, professor]).   % professor(X):-philosopher(X)


p(6,[no,vegetarian,is,a,meat_eater]).  % false:-meat_eater(X),vegetarian(X)
p(6,[some,vegetarian,is,a,policeman]).
    % vegetarian(some(vegetarian,policeman)):-true
    % policeman(some(vegetarian,policeman)):-true


p(7,[some,student,is,rich]).
   %  student(some(student,rich)):- true
   % rich(some(student,rich)):- true
p(7,[some,girl,is,not,a,student]).
   % girl(some(girl,not(student))):-true
   % false:-student(some(girl,not(student))

p(8,[some,man,is,a,clown]).
   % man(some(man,clown)):-true
   % clown(some(man,clown)):-true
p(8,[some,man,is,bald]).
   % man(some(man,bald)):-true
   % bald(some(man,bald):-true



c(1,[no,robin,is,a,reptile]).
%     Opposite is:  [some,robin,is,a,reptile]).
%     robin(some(robin,reptile)):-true
%     reptile(some(robin,reptile):-true
% false can be derived, syllogism is valid


c(2,[a,human,is,warm_blooded]).
%     Opposite is: [some,human,is,not,warm_blooded]).
%     human(some(human,not(warm_blooded))):-true
%     false:-warm_blooded(some(human,not(warm_blooded)))
% false can be derived, syllogism is valid


c(3,[some,mammal,is,deaf]).
%      Opposite is: [no,mammal,is,deaf]).
%      false:-mammal(X),deaf(X))
% false can be derived, syllogism is valid


c(4,[no,reptile,is,a,robin]).
    %  Opposite is: [some,reptile,is,a,robin]).
    % robin(some(reptile,robin)):-true
        % reptile(some(reptile,robin)):-true
% false can be derived, syllogism is valid

c(5,[every,logician,is,a,professor]).
      % Opposite is [some,logician,is,not,a,professor]
      % logician(some(logician,not(professor))):-true
      % false:-professor(some(logician,not(professor)))
% False cannot be derived, syllogism is invalid

c(6,[some,policeman,is,not,a,meat_eater]).
  % Negated conclusion is: [a,policeman,is,a,meat_eater]
  % meat_eater(X):-policeman(X)
% false can be derived, syllogism is valid


c(7,[some,girl,is,not,rich]).
  %  Opposites: [a,girl,is,rich]
  % rich(X):-girl(X)
% false cannot be derived, syllogism is invalid

c(8,[some,clown,is,bald]).
  % Opposite is [no,clown,is,bald]
  % false:-clown(X),bald(X)
% false cannot be derived, syllogism is invalid
% compiling /root/labcat/labcat/engines/lib/prolog/automarker.pl...
%  loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/timeout.po...
%  module timeout imported into user
%   loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/types.po...
%   module types imported into timeout
%   loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/types.po in module types, 0 msec 4112 bytes
%   loading foreign resource /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/x86_64-linux-glibc2.17/timeout.so in module timeout
%  loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/timeout.po in module timeout, 0 msec 51056 bytes
%  loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/file_systems.po...
%  module file_systems imported into user
%   module types imported into file_systems
%   loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/system.po...
%   module system imported into file_systems
%    module types imported into system
%   loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/system.po in module system, 10 msec 4720 bytes
%   loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/lists.po...
%   module lists imported into file_systems
%    module types imported into lists
%   loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/lists.po in module lists, 0 msec 107408 bytes
%  loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/file_systems.po in module file_systems, 10 msec 165488 bytes
% compiled /root/labcat/labcat/engines/lib/prolog/automarker.pl in module user, 230 msec 1220880 bytes
SICStus 4.3.5 (x86_64-linux-glibc2.17): Tue Dec  6 10:41:06 PST 2016
Licensed to SP4.3doc.ic.ac.uk
% compiling /tmp/d20181122-38-97zyad/src/solution.pl...
% compiled /tmp/d20181122-38-97zyad/src/solution.pl in module autom, 10 msec 30480 bytes
yes
% compiling /tmp/d20181122-38-97zyad/src/utilities.pl...
% compiled /tmp/d20181122-38-97zyad/src/utilities.pl in module autom, 0 msec 2512 bytes
yes
% compiling /tmp/d20181122-38-97zyad/src/private_test_data.pl...
% compiled /tmp/d20181122-38-97zyad/src/private_test_data.pl in module autom, 10 msec 3024 bytes
yes
% compiling /tmp/d20181122-38-97zyad/src/arguments.pl...
% compiled /tmp/d20181122-38-97zyad/src/arguments.pl in module autom, 0 msec 7856 bytes
yes
% compiling /tmp/d20181122-38-97zyad/src/arguments.pl...
% compiled /tmp/d20181122-38-97zyad/src/arguments.pl in module submission, 0 msec 7440 bytes
* /tmp/d20181122-38-97zyad/src/arguments.pl is previously loaded into module autom
*           now it is reloaded into module submission
yes
% compiling /tmp/d20181122-38-97zyad/src/utilities.pl...
% compiled /tmp/d20181122-38-97zyad/src/utilities.pl in module submission, 0 msec 3488 bytes
* /tmp/d20181122-38-97zyad/src/utilities.pl is previously loaded into module autom
*           now it is reloaded into module submission
yes
% compiling /tmp/d20181122-38-97zyad/src/syllogisms.pl...
%  including /tmp/d20181122-38-97zyad/src/arguments.pl...
The procedure p/2 is being redefined.
    Old file: /tmp/d20181122-38-97zyad/src/arguments.pl
    New file: /tmp/d20181122-38-97zyad/src/syllogisms.pl
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don't ask again
       s    don't redefine this procedure and don't ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
The procedure c/2 is being redefined.
    Old file: /tmp/d20181122-38-97zyad/src/arguments.pl
    New file: /tmp/d20181122-38-97zyad/src/syllogisms.pl
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don't ask again
       s    don't redefine this procedure and don't ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don't ask again
       s    don't redefine this procedure and don't ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don't ask again
       s    don't redefine this procedure and don't ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
%  included /tmp/d20181122-38-97zyad/src/arguments.pl in module submission, 0 msec 33472 bytes
%  including /tmp/d20181122-38-97zyad/src/utilities.pl...
The procedure forall/2 is being redefined.
    Old file: /tmp/d20181122-38-97zyad/src/utilities.pl
    New file: /tmp/d20181122-38-97zyad/src/syllogisms.pl
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don't ask again
       s    don't redefine this procedure and don't ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don't ask again
       s    don't redefine this procedure and don't ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don't ask again
       s    don't redefine this procedure and don't ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don't redefine this procedure
       p    redefine this procedure and don’t ask again
       s    don’t redefine this procedure and don’t ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
       y    redefine this procedure
       n    don’t redefine this procedure
       p    redefine this procedure and don’t ask again
       s    don’t redefine this procedure and don’t ask again
       a    abort
       b    break
       ?    print this information
Do you really want to redefine it?  (y, n, p, s, a, b, or ?)
%  included /tmp/d20181122-38-97zyad/src/utilities.pl in module submission, 0 msec 1472 bytes
% compiled /tmp/d20181122-38-97zyad/src/syllogisms.pl in module submission, 20 msec 55760 bytes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
yes
```

```
---------- Test 1 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([a,cat,is,a,mammal],Opp).
  Opp = [some,cat,is,not,a,mammal] ;         %% correct

---------- Test 2 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([every,cat,is,a,mammal],Opp).
  Opp = [some,cat,is,not,a,mammal] ;         %% correct

---------- Test 3 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([some,cat,is,a,mammal],Opp).
  Opp = [no,cat,is,a,mammal] ;         %% correct

---------- Test 4 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([some,cat,is,not,a,mammal],Opp).
  Opp = [a,cat,is,a,mammal] ;         %% correct

---------- Test 5 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([a,banker,is,greedy],Opp).
  Opp = [some,banker,is,not,greedy] ;         %% correct

---------- Test 6 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([every,banker,is,greedy],Opp).
  Opp = [some,banker,is,not,greedy] ;         %% correct

---------- Test 7 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([no,banker,is,greedy],Opp).
  Opp = [some,banker,is,greedy] ;         %% correct

---------- Test 8 ::  Opposite -------------------------------------------------

| ? find Opp: opposite([some,banker,is,not,greedy],Opp).
  Opp = [a,banker,is,greedy] ;         %% correct

================================================================================
Step 2 (syllogism/1)

---------- Test 9 ::  Parsing Syllogisms ---------------------------------------

| ? find Clauses: syllogism(Clauses,[a,cat,is,a,mammal],[]).
  Clauses = [(mammal(_3173):-cat(_3173))] ;
No more solutions    (All correct)
No missing solutions

---------- Test 10 ::  Parsing Syllogisms --------------------------------------

| ? find Clauses: syllogism(Clauses,[every,cat,is,a,mammal],[]).
  Clauses = [(mammal(_3101):-cat(_3101))] ;
No more solutions    (All correct)
No missing solutions

---------- Test 11 ::  Parsing Syllogisms --------------------------------------

| ? find Clauses: syllogism(Clauses,[some,cat,is,a,mammal],[]).
  Clauses = [(cat(some(cat,mammal)):-true),(mammal(some(cat,mammal)):-true)] ;
No more solutions    (All correct)
No missing solutions

---------- Test 12 ::  Parsing Syllogisms --------------------------------------

| ? find Clauses: syllogism(Clauses,[some,cat,is,not,a,mammal],[]).
  Clauses = [(cat(some(cat,not(mammal))):-true),(false:-mammal(some(cat,not(mammal))))] ;
No more solutions    (All correct)
No missing solutions

---------- Test 13 ::  Parsing Syllogisms --------------------------------------

| ? find Clauses: syllogism(Clauses,[no,banker,is,greedy],[]).
  Clauses = [(false:-banker(_3097),greedy(_3097))] ;
No more solutions    (All correct)
No missing solutions


================================================================================
Step 2 (syllogism/1)
TESTS PASSED:  13 / 13 ;  MARKS:  38 / 38
================================================================================

================================================================================
Step 3 (translate/1, cl/3)


About to call translate(1) ... [[a,robin,is,a,bird],[no,bird,is,a,reptile]]
[(false:-bird(_4227),reptile(_4227)),(bird(_4109):-robin(_4109)),(robin(some(robin,reptile)):-true),(reptile(some(robin,reptile)):-true)]done
About to call translate(2) ... [[a,human,is,a,mammal],[a,mammal,is,warm_blooded]]
[(warm_blooded(_4121):-mammal(_4121)),(mammal(_4003):-human(_4003)),(human(some(human,not(warm_blooded))):-true),(false:-warm_blooded(some(human,not(warm_blooded))))]done
About to call translate(3) ... [[a,human,is,a,mammal],[some,human,is,deaf]]
[(human(some(human,deaf)):-true),(deaf(some(human,deaf)):-true),(mammal(_3945):-human(_3945)),(false:-mammal(_3821),deaf(_3821))]done
About to call translate(4) ... [[a,robin,is,a,bird],[no,bird,is,a,reptile]]
[(false:-bird(_4109),reptile(_4109)),(bird(_3991):-robin(_3991)),(reptile(some(reptile,robin)):-true),(robin(some(reptile,robin)):-true)]done
About to call translate(5) ... [[every,philosopher,is,a,logician],[every,philosopher,is,a,professor]]
[(professor(_4143):-philosopher(_4143)),(logician(_4025):-philosopher(_4025)),(logician(some(logician,not(professor))):-true),(false:-professor(some(logician,not(professor))))]done
About to call translate(6) ... [[no,vegetarian,is,a,meat_eater],[some,vegetarian,is,a,policeman]]
[(vegetarian(some(vegetarian,policeman)):-true),(policeman(some(vegetarian,policeman)):-true),(false:-vegetarian(_3985),meat_eater(_3985)),(meat_eater(_3867):-policeman(_3867))]done
About to call translate(7) ... [[some,student,is,rich],[some,girl,is,not,a,student]]
[(girl(some(girl,not(student))):-true),(false:-student(some(girl,not(student)))),(student(some(student,rich)):-true),(rich(some(student,rich)):-true),(rich(_3851):-girl(_3851))]done
About to call translate(8) ... [[some,man,is,a,clown],[some,man,is,bald]]
[(man(some(man,bald)):-true),(bald(some(man,bald)):-true),(man(some(man,clown)):-true),(clown(some(man,clown)):-true),(false:-clown(_3821),bald(_3821))]done

---------- Test 1 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(1,_1185,_1225),Cl=(_1185:-_1225).
  Cl = false:-bird(_3611),reptile(_3611) ;
  Cl = bird(_3605):-robin(_3605) ;
  Cl = robin(some(robin,reptile)):-true ;
  Cl = reptile(some(robin,reptile)):-true ;
No more solutions    (All correct)
No missing solutions

---------- Test 2 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(2,_1185,_1225),Cl=(_1185:-_1225).
  Cl = warm_blooded(_3643):-mammal(_3643) ;
  Cl = mammal(_3643):-human(_3643) ;
  Cl = human(some(human,not(warm_blooded))):-true ;
  Cl = false:-warm_blooded(some(human,not(warm_blooded))) ;
No more solutions    (All correct)
No missing solutions

---------- Test 3 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(3,_1185,_1225),Cl=(_1185:-_1225).
  Cl = human(some(human,deaf)):-true ;
  Cl = deaf(some(human,deaf)):-true ;
  Cl = mammal(_3643):-human(_3643) ;
  Cl = false:-mammal(_3649),deaf(_3649) ;
No more solutions    (All correct)
No missing solutions

---------- Test 4 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(4,_1185,_1225),Cl=(_1185:-_1225).
  Cl = false:-bird(_3653),reptile(_3653) ;
  Cl = bird(_3647):-robin(_3647) ;
  Cl = reptile(some(reptile,robin)):-true ;
  Cl = robin(some(reptile,robin)):-true ;
No more solutions    (All correct)
No missing solutions

---------- Test 5 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(5,_1185,_1225),Cl=(_1185:-_1225).
  Cl = professor(_3647):-philosopher(_3647) ;
  Cl = logician(_3647):-philosopher(_3647) ;
  Cl = logician(some(logician,not(professor))):-true ;
  Cl = false:-professor(some(logician,not(professor))) ;
No more solutions    (All correct)
No missing solutions

---------- Test 6 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(6,_1185,_1225),Cl=(_1185:-_1225).
  Cl = vegetarian(some(vegetarian,policeman)):-true ;
  Cl = policeman(some(vegetarian,policeman)):-true ;
  Cl = false:-vegetarian(_3657),meat_eater(_3657) ;
  Cl = meat_eater(_3651):-policeman(_3651) ;
No more solutions    (All correct)
No missing solutions

---------- Test 7 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(7,_1185,_1225),Cl=(_1185:-_1225).
  Cl = girl(some(girl,not(student))):-true ;
  Cl = false:-student(some(girl,not(student))) ;
  Cl = student(some(student,rich)):-true ;
  Cl = rich(some(student,rich)):-true ;
  Cl = rich(_3647):-girl(_3647) ;
No more solutions    (All correct)
No missing solutions

---------- Test 8 ::  Asserting Translations -----------------------------------

| ? find Cl: cl(8,_1185,_1225),Cl=(_1185:-_1225).
  Cl = man(some(man,bald)):-true ;
  Cl = bald(some(man,bald)):-true ;
  Cl = man(some(man,clown)):-true ;
  Cl = clown(some(man,clown)):-true ;
  Cl = false:-clown(_3649),bald(_3649) ;
No more solutions    (All correct)
No missing solutions


================================================================================
Step 3 (translate/1, cl/3)
TESTS PASSED:  8 / 8 ;  MARKS:  12 / 12
================================================================================

================================================================================
Step 4 (eval/2, valid/1, invalid/1)

---------- Test 1 ::  Evaluate An Argument -------------------------------------

| ? eval(1,false).
yes        %% correct

---------- Test 2 ::  Evaluate An Argument -------------------------------------

| ? eval(2,false).
yes        %% correct

---------- Test 3 ::  Evaluate An Argument -------------------------------------

| ? eval(3,false).
yes        %% correct

---------- Test 4 ::  Evaluate An Argument -------------------------------------

| ? eval(4,false).
yes        %% correct

---------- Test 5 ::  Evaluate An Argument -------------------------------------

| ? eval(5,false).
no         %% correct

---------- Test 6 ::  Evaluate An Argument -------------------------------------

| ? eval(6,false).
yes        %% correct

---------- Test 7 ::  Evaluate An Argument -------------------------------------

| ? eval(7,false).
no         %% correct

---------- Test 8 ::  Evaluate An Argument -------------------------------------

| ? eval(8,false).
no         %% correct

---------- Test 9 ::  Find 5 Valid ---------------------------------------------

| ? find N: valid(N).
  N = 1 ;
  N = 2 ;
  N = 3 ;
  N = 4 ;
  N = 6 ;
No more solutions    (All correct)
No missing solutions

---------- Test 10 ::  Valid? Not an Argument ---------------------------------

| ? valid(p0).
no         %% correct

---------- Test 11 ::  Find 3 Invalid ------------------------------------------
```

```

| ? find N: invalid(N).
No solution         %% WRONG

Model answer produces the following solutions:
  N = 5 ;
  N = 7 ;
  N = 8 ;
Model solutions end

---------- Test 12 ::  Invalid? Not an Argument -------------------------------

| ? invalid(p0).
yes        %% WRONG


================================================================================
Step 4 (eval/2, valid/1, invalid/1)
TESTS PASSED:  10 / 12 ;  MARKS:  13 / 17
================================================================================

================================================================================
Step 5 (test/1)

---------- Test 1 ::  Evaluate and Output Argument ----------------------------

syllogism 1:
   a robin is a bird
   no bird is a reptile
   =>
   no robin is a reptile

Premises and opposite of conclusion converted to clauses:
   false:-bird(_3053),reptile(_3053).
   bird(_3047):-robin(_3047).
   robin(some(robin,reptile)):-true.
   reptile(some(robin,reptile)):-true.

false can be derived, syllogism 1 is valid.


---------- Test 2 ::  Evaluate and Output Argument ----------------------------

syllogism 2:
   a human is a mammal
   a mammal is warm_blooded
   =>
   a human is warm_blooded

Premises and opposite of conclusion converted to clauses:
   warm_blooded(_2983):-mammal(_2983).
   mammal(_2983):-human(_2983).
   human(some(human,not(warm_blooded))):-true.
   false:-warm_blooded(some(human,not(warm_blooded))).

false can be derived, syllogism 2 is valid.


---------- Test 3 ::  Evaluate and Output Argument ----------------------------

syllogism 3:
   a human is a mammal
   some human is deaf
   =>
   some mammal is deaf

Premises and opposite of conclusion converted to clauses:
   human(some(human,deaf)):-true.
   deaf(some(human,deaf)):-true.
   mammal(_2983):-human(_2983).
   false:-mammal(_2989),deaf(_2989).

false can be derived, syllogism 3 is valid.


---------- Test 4 ::  Evaluate and Output Argument ----------------------------

syllogism 4:
   a robin is a bird
   no bird is a reptile
   =>
   no reptile is a robin

Premises and opposite of conclusion converted to clauses:
   false:-bird(_3017),reptile(_3017).
   bird(_3011):-robin(_3011).
   reptile(some(reptile,robin)):-true.
   robin(some(reptile,robin)):-true.

false can be derived, syllogism 4 is valid.


---------- Test 5 ::  Evaluate and Output Argument ----------------------------

syllogism 5:
   every philosopher is a logician
   every philosopher is a professor
   =>
   every logician is a professor

Premises and opposite of conclusion converted to clauses:
   professor(_3011):-philosopher(_3011).
   logician(_3011):-philosopher(_3011).
   logician(some(logician,not(professor))):-true.
   false:-professor(some(logician,not(professor))).

false cannot be derived, syllogism 5 is invalid.


---------- Test 6 ::  Evaluate and Output Argument ----------------------------

syllogism 6:
   no vegetarian is a meat_eater
   some vegetarian is a policeman
   =>
   some policeman is not a meat_eater

Premises and opposite of conclusion converted to clauses:
   vegetarian(some(vegetarian,policeman)):-true.
   policeman(some(vegetarian,policeman)):-true.
   false:-vegetarian(_3033),meat_eater(_3033).
   meat_eater(_3027):-policeman(_3027).

false can be derived, syllogism 6 is valid.


---------- Test 7 ::  Evaluate and Output Argument ----------------------------

syllogism 7:
   some student is rich
   some girl is not a student
   =>
   some girl is not rich

```

```
Premises and opposite of conclusion converted to clauses:
   girl(some(girl,not(student))):-true.
   false:-student(some(girl,not(student))).
   student(some(student,rich)):-true.
   rich(some(student,rich)):-true.
   rich(_3011):-girl(_3011).

false cannot be derived, syllogism 7 is invalid.


---------- Test 8 ::  Evaluate and Output Argument -----------------------------

syllogism 8:
   some man is a clown
   some man is bald
   =>
   some clown is bald

Premises and opposite of conclusion converted to clauses:
   man(some(man,bald)):-true.
   bald(some(man,bald)):-true.
   man(some(man,clown)):-true.
   clown(some(man,clown)):-true.
   false:-clown(_2989),bald(_2989).

false cannot be derived, syllogism 8 is invalid.



================================================================================
Step 5 (test/1)
TESTS PASSED:  8 / 8 ;  MARKS:  8 / 8
================================================================================


================================ SUMMARY (???) =================================

TESTS PASSED:  39 / 41 ;  MARKS:  71 / 75

================================================================================
```